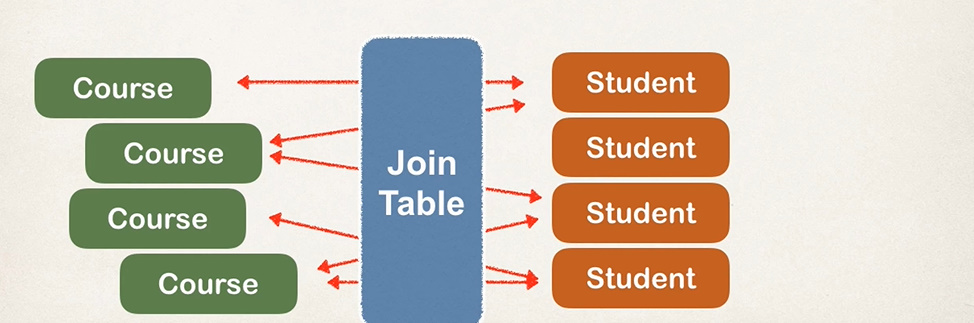
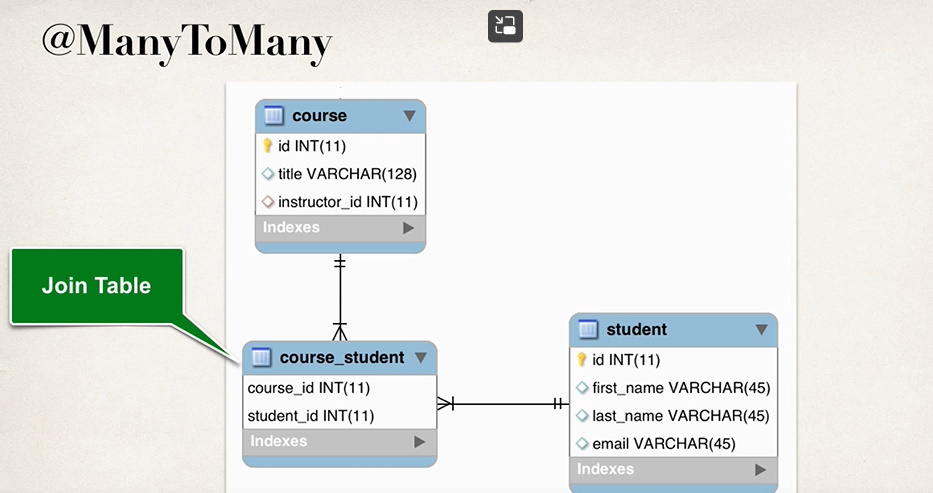
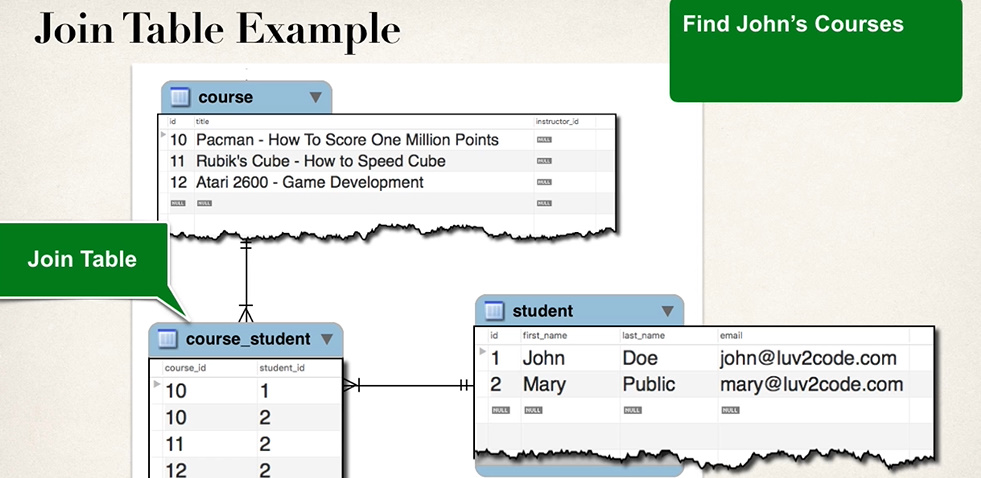
**ManyToMany**

* In ManyToMany fiecare obiect ar trebui sa aiba o lista catre toate obiectele ce-i corespund.
* Facem exact asa cum se cere atunci cand avem relatii ManyToMany la crearea tabelelor. Adica, cream un al 3 tabel, unde drept primary key vor fi primary key de la fiecare tabel. Totodata, acestea vor fi foreign si key in tabelul nou.

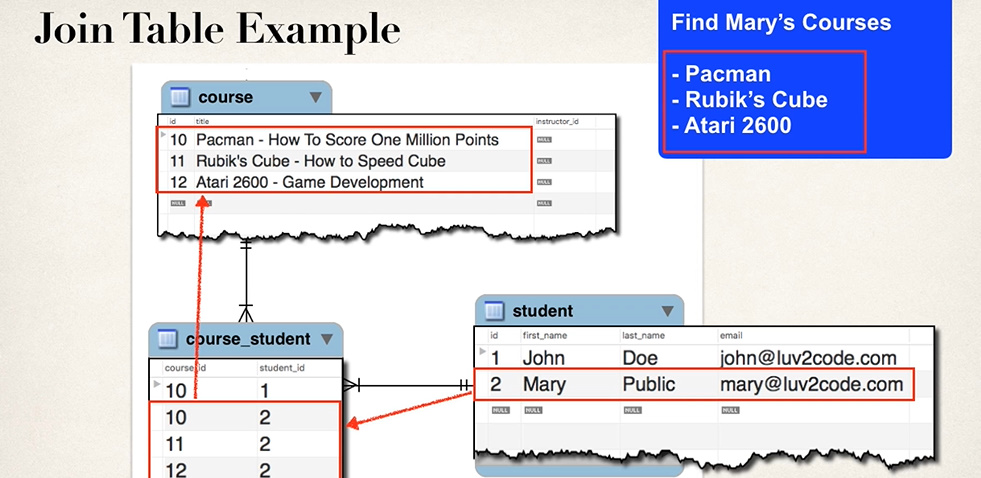


* Acest tabel e numit Join Table. El are foreign key pentru fiecare tabel si creaza o mapare pentru ambele tabele.





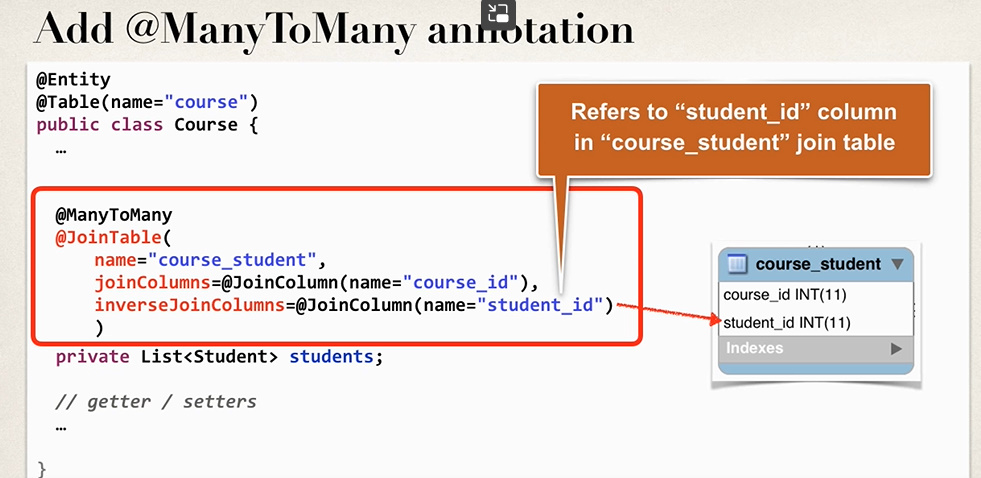
* De ex:



Daca vrem sa aflam cursurile in care e inscrisa Mary, vom cauta in Join Table unde student\_id e egal cu al ei, adica 2, si pentru toate randurile care au student id = 2 adaugam cursurile din tabelul de Cursuri cu id corespunzator.

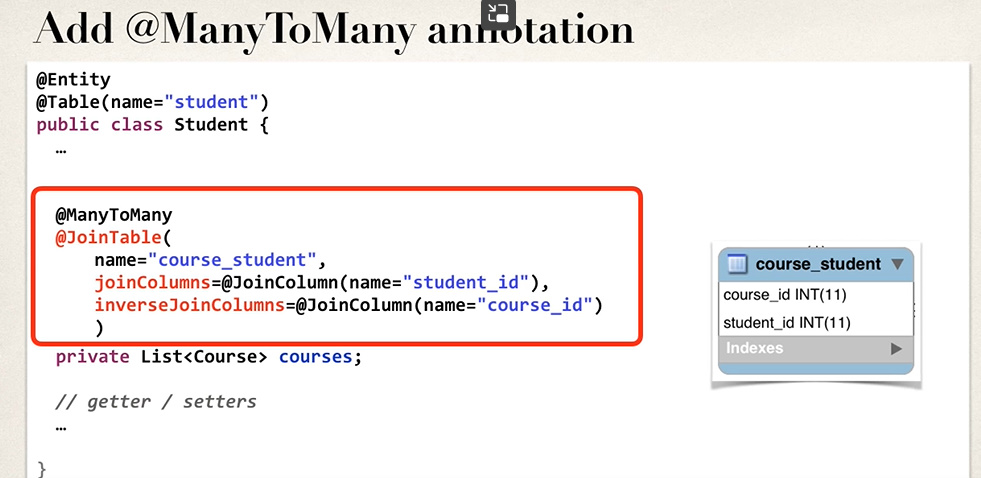
* Clasa Course va avea o lista cu Student, si clasa Student o lista cu Course

**Definirea relatie in clase**

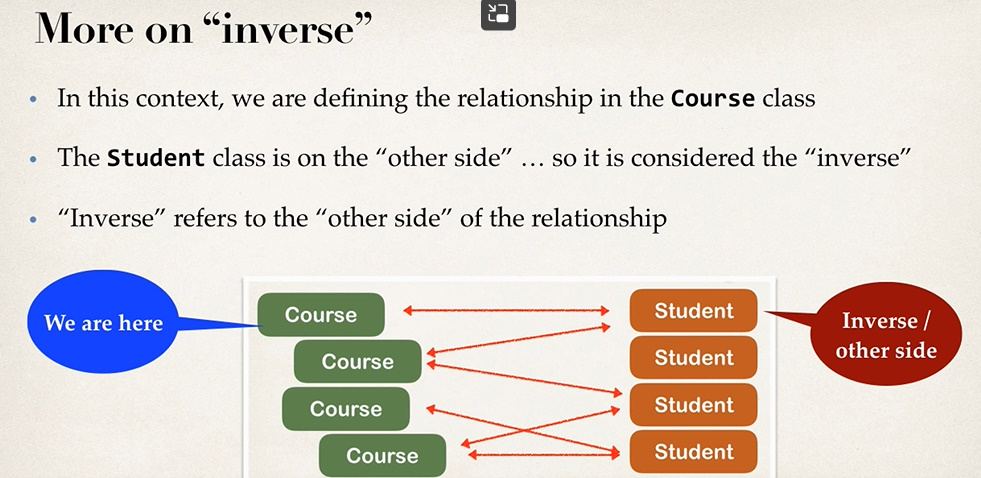
* Nu trebuie sa cream o Entity speciala pentru acest Join Table, asa cum el contine doar foreign keys.
* 

Deci, cand folosim @ManyToMany, trebuie mereu sa specificam care este @JoinTable. @JoinTable cere specificarea numelui tabelului.

* JoinColumns arata care e numele coloanei/coloanelor in tabel care e legata de tabelul course.Deci, joinColumns=@JoinColumn(name=”course\_id”) spune ca in tabelul course\_student, foreign key este primary key din tabelul course,si aceasta coloana din Join Table se leaga de coloana cu numele student\_id care se refera la primary key din alt tabel. Hibernate va sti ca course\_id foreign key se va lega de primary key din tabelul course, caci doar suntem in entitatea Course. Asa, el va folosi un course\_id pentru a gasi toate student\_id, de ex, pentru acesta. Deci, el va extrage obiectul din curs tabel, pe baza la course\_id din Join Tabel si apoi va extrage obiectele din student tabel
* Acum, facem la fel si pentru Student Entity, doar ca deja setam invers:

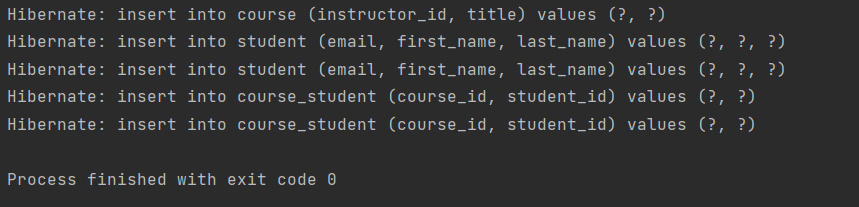


**De ce e inversJoinColumn?**



Spunem ca e invers, deoarece joinColumns creaza legatura intr-o directie, adica,pentru clasa Course, de la prima clasa la a 2, si inverseJoinColumn in directia opusa, adica de la a 2 clasa la prima. Pentru a 2 clasa va fi invers.

**Adaugarea obiectelor**

* Cand vom adauga obiecte, putem utiliza Lista, asa cum face deobicei. Fie ca cream un curs si o lista de studenti in el, fie invers, nu conteaza.
* Course course1 = new Course("Hibernate");  
    
  Student student1 = new Student("Mititiuc","Eduard","test@mail.ru");  
  Student student2 = new Student("Goncear","Alex","mail@mail.ru");  
  course1.addStudent(student1);  
  course1.addStudent(student2);  
  session.save(course1);
* Mai sus nu uitam de @Cascade
* 

Hibernate intai de toate a salvat cursul in tabelul course, apoi studentii in tabelul student si abea la final a adaugat primary key a celor 3 obiecte in tabelul student\_course

* Mai sus am adaugat un curs si in el am adaugat studenti. Putem face asta si invers, oricum totul e la fel. Putem selecta si studentul din baza de date si adauga cursuri pentru el:

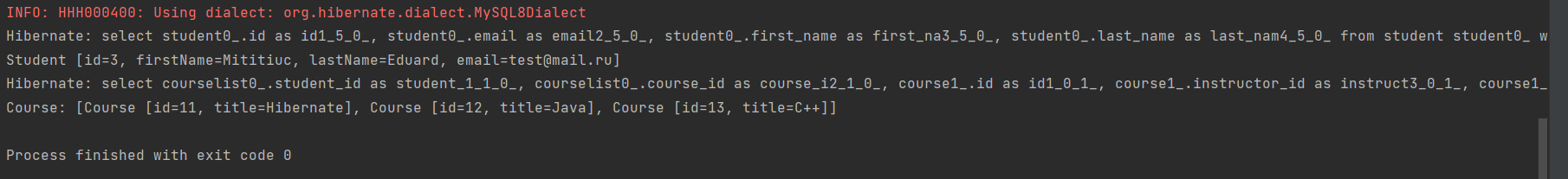
Student student = session.get(Student.class,3);  
  
System.*out*.println(student);  
System.*out*.println("Course: "+student.getCourseList());  
  
Course course1 = new Course("Java");  
Course course2 = new Course("C++");  
student.addCourse(course1);  
student.addCourse(course2);  
  
session.saveOrUpdate(student);  
session.getTransaction().commit();

**Obtinerea obiectelor din baza de date**

* Obtinereaa obiectelor se face ca deobicei:

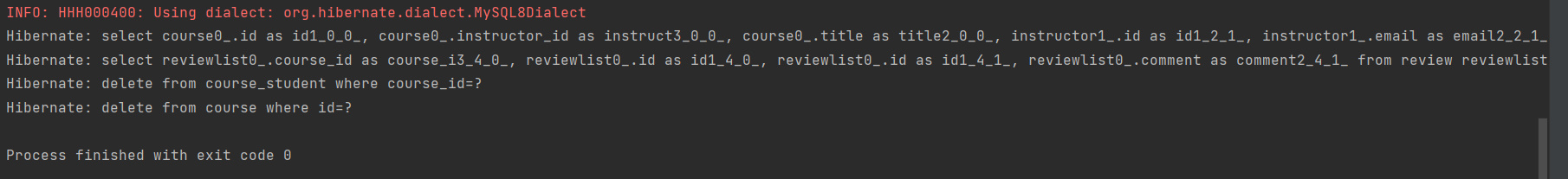
session.beginTransaction();  
 Student student = session.get(Student.class,3);  
  
System.*out*.println(student);  
System.*out*.println("Course: "+student.getCourseList());  
  
session.getTransaction().commit();

* Dupa ce obtine obiectul Student, Hibernate se foloseste de JoinTable pentru depista tuplurile care au id al student identic cu acesta si apoi ia pentru fiecare tuplu gasit id al course, se duce in tabelul course si prea obiectele cu id corespunzatoar



**Stergerea obiectelor**

session.beginTransaction();  
 Course course = session.get(Course.class,11);  
 session.delete(course);  
  
session.getTransaction().commit();



Hibernate intai sterge cursul din tabelul Join si apoi din tabelul course, deoarece tabelul Join e cel ce are foreign key, si daca se sterge intai din course, iese ca pana se sterge cursul si din Join tabel, exista o foreign key invalida.